Aula 04

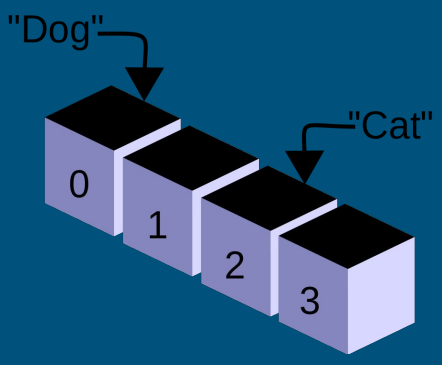
Introdução à Programação

emerson@paduan.pro.br

Antes de iniciar...

Dúvidas ???

Vetores



Armazenando como conjunto.

emerson@paduan.pro.br

O que é ?

Estrutura de Dados capaz de armazenar um conjunto de dados de um mesmo tipo (Homogêneo).

É uma estrutura de tamanho fixo (estático) na criação.

emerson@paduan.pro.br

Aplicação

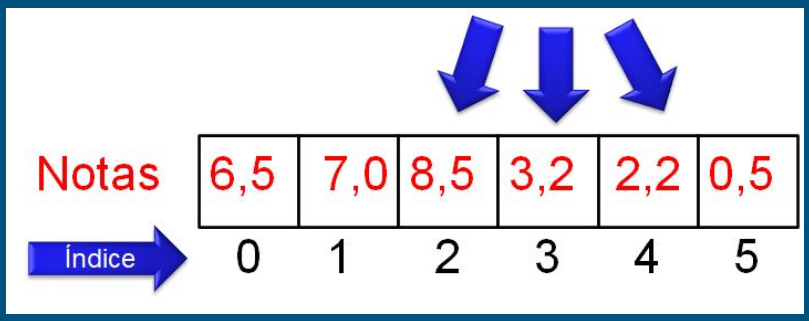
Como armazenar as notas de dez alunos?

Opções:

Declarar 10 variáveis

Criar um vetor com 10 posições

Visual



em Java

Um vetor é declarado definindo-se seu nome (*plural*), tipo e tamanho.

```
int [ ] idades;    // declara um vetor chamado idades para armazenar inteiros

idades = new  int [10];    // aloca (cria) o vetor com 10 posições
```

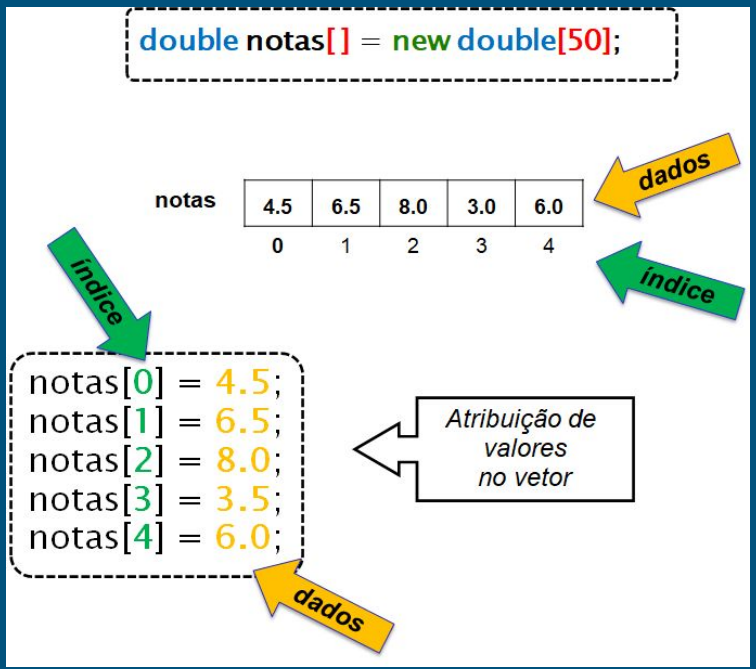
Pode ser feito:

```
int [ ] idades = new int[10];
```

Você também encontrará:

```
int idades[ ] = new int[10];
```


usando o índice



Para saber o tamanho do vetor é possível usar:

vetor.length

Exemplo 01

Lendo um conjunto de notas e exibindo os dados do vetor:

```
for( i = 0; i < notas.length; i++){
      System.out.println("Digite uma nota:");
      notas[ i ] = entrada.nextDouble();
}
```

```
for( i = 0; i < 5; i++){
     System.out.println("Nota: " + notas[ i ]);
}
```

4.5	6.5	8.0	3.5	6.0
0	1	2	3	4

```
Nota: 4.5
Nota: 6.5
Nota: 8.0
Nota: 3.5
Nota: 6.0
CONSTRUÍDO COM SUCESSO
```


Exercício 4-1

Escreva um programa que leia as notas da prova de 10 alunos e armazene em um vetor.

A seguir, mostre:

Todas as notas, a média da turma, a quantidade de notas acima da média da turma.

Exercício 4-2

Escreva um programa que leia os nomes de 7 pessoas e armazene em um vetor.

A seguir, mostre os nomes em ordem inversa ao que foram digitados.

Exemplo:

Digitado: Marcos João . . . Samanta Felipe

Exibido: Felipe Samanta . . . João Marcos

emerson@paduan.pro.br

Exercício 4-3

O dono de um cassino deseja saber se o dado usado em uma mesa está "viciado". Para testar o dado, ele quer jogar o dado 100 vezes e verificar se cada face do dado ocorreu com a mesma frequência.

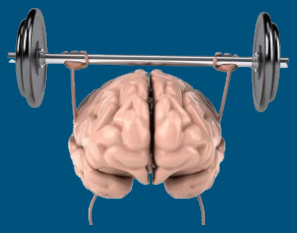
Escreva um programa para ajudar com esta tarefa.

Exemplo:
O lado 1 foi sorteado 17 vezes
O lado 2 foi sorteado 16 vezes
.... etc.

OBS: Para fazer a simulação, utilize a geração de números aleatórios para simular o lançamento do dado e não precisar digitar os 100 valores.

emerson@paduan.pro.br

Exercício 4-3

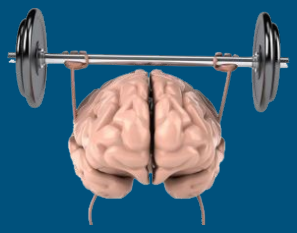


```
import java.util.Random;

....

Random random = new Random();
```

```
int sorteado;

//sorteia um inteiro (dentro do range de inteiros)
sorteado = random.nextInt();

//sorteia um inteiro entre 0 e TAM-1
sorteado = random.nextInt( 10 );

//sorteia um inteiro entre 0 e TAM-1 e
//soma 1 ao valor sorteado
sorteado = 1 + random.nextInt( 10 ); //entre 1 e 10
```

emerson@paduan.pro.br

Exercício 4-4 (desafio)



Escreva um programa que dado um valor numérico digitado pelo usuário (armazenado em uma variável inteira), imprima cada um dos seus dígitos por extenso.

Exemplo:

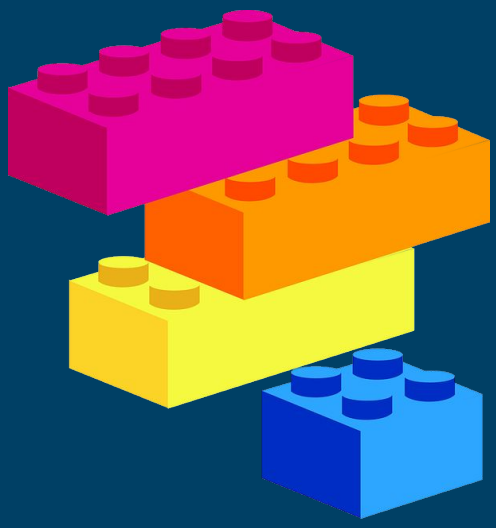
Entrada: 4571

Saída: quatro, cinco, sete, um

emerson@paduan.pro.br

Modularização

Antes de POO...



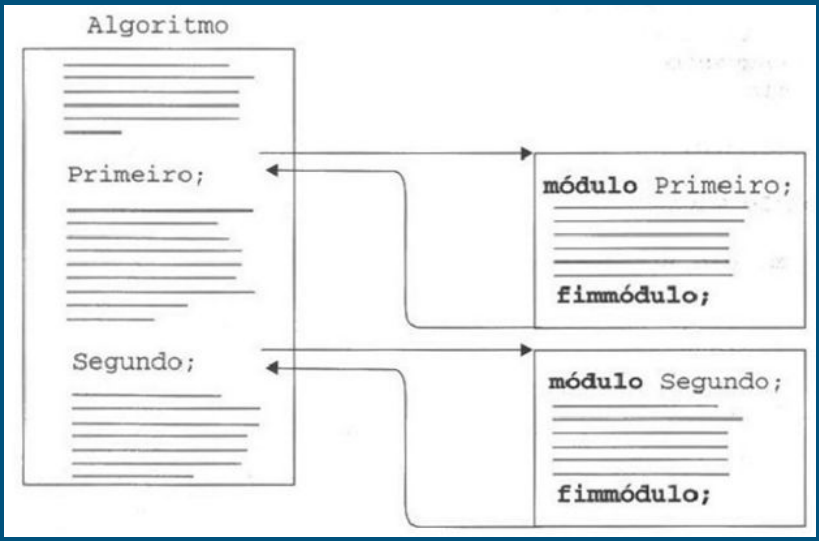
O que é ?

- Modularizar é dividir um programa em Sub-rotinas, chamadas de Procedimentos ou Funções. Estes são blocos de programa que executam determinada tarefa.

*** Na Programação Orientada a Objetos chamamos Métodos

emerson@paduan.pro.br

O que é ?



Para quê ?

Algumas vantagens:

- Dividir e estruturar o problema em pequenas partes para facilitar o desenvolvimento;
- Evitar repetição de código em vários locais;
- Facilitar a localização e correção de problemas;
- Facilitar a manutenção do código;

Sintaxe básica

```
Tipo-de-retorno NomeDoMétodo ( lista de parâmetros ) {
  //corpo do módulo
  retorno
}
```

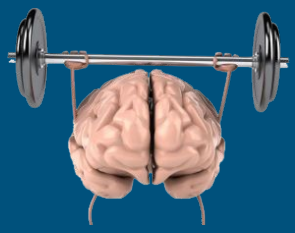
- Nome do método: Valem as regras de nome de variáveis
- Lista de parâmetros: Opcional. Tipos e nomes das variáveis que o método irá receber.
- Corpo: Instruções que realizam a operação pretendida.
- Tipo de Retorno: Valor que o método retorna caso exista.

Praticando

Entendendo por meio de exemplos:

```
void linha();

void linha(int );

void linha(int, char);

int soma (int, int);
```

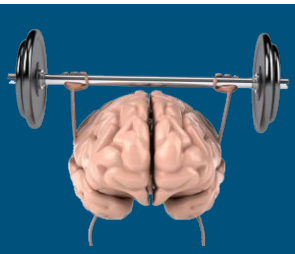
Exercício 4-5



Escreva uma função em Java que encontre o menor entre três números fornecidos como parâmetros.

emerson@paduan.pro.br

Exercício 4-6



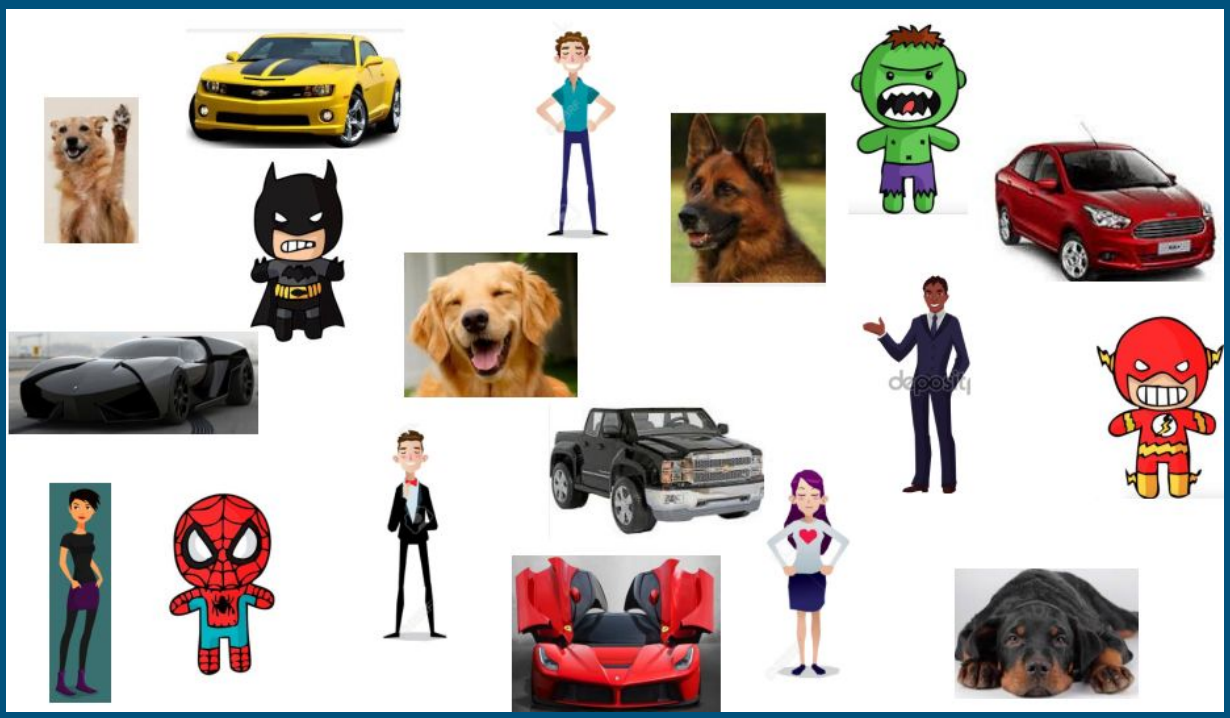
Escreva uma função em Java que conte quantas vogais existem um uma palavra dada como parâmetro.

emerson@paduan.pro.br

POO

Programação Orientada a Objetos

emerson@paduan.pro.br



Como organizar ?

emerson@paduan.pro.br

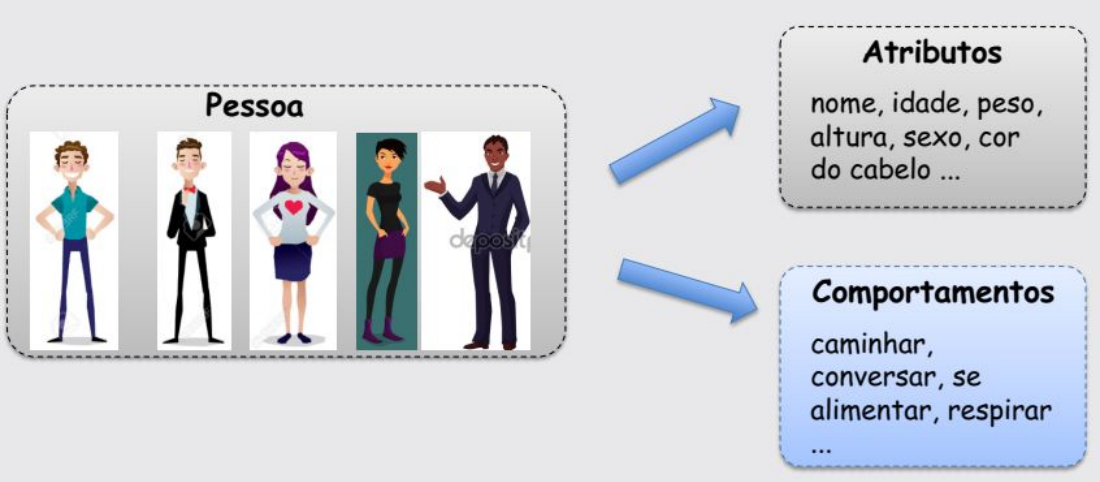
Características comuns

Classe

Uma classe é um modelo que define, especifica um objeto. É uma abstração (representação) dos objetos.

Ela define os dados (ATRIBUTOS) e os comportamentos (MÉTODOS) do Objeto.

Exemplo



ATENÇÃO

Uma classe é um **MODELO**!

Não se coloca dados ou se utiliza diretamente uma classe.

É necessário criar um objeto a partir da classe

Exemplo

TODOS os objetos criados a partir da classe, possuem os mesmos atributos e métodos, mas com valores diferentes.

Exemplo

```
class Pessoa {
        String nome;

        void apresentar(){
                System.out.println(“Olá! Eu sou ” + nome);
        }
}
```

```
class Exemplo{
        main ( ) {

        Pessoa p = new Pessoa( );

        p.nome = "Emerson";
        p.apresentar( );
}
```

Let's code

Criando a classe Livro

Construtores

São métodos especiais utilizados para inicialização dos atributos de um objeto no “momento” da criação do objeto.

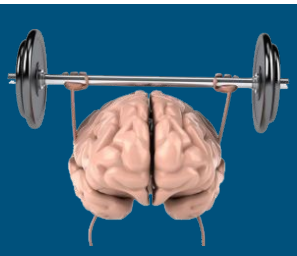
Detalhes:
Os construtores podem ter ou não parâmetros
O nome do construtor DEVE ser o mesmo da classe
Construtores NÃO possuem valor de retorno

Exemplo

```
public class Pessoa {
   String nome;
   float salario;

   public Pessoa(String nome, float salario){
      this.nome = nome;
      this.salario = sal;
   }

   public void exibir(){
      System.out.println("Pessoa: " + nome + ": R$ " + salario);
   }
}
```

```
class Exemplo{
        main ( ) {

        Pessoa p = new Pessoa("Marcos", 5000 );

        p.exibir( );
 }
```


Exercício 4-7



Crie a classe veículo, com os atributos modelo, marca e consumo (quantos km/l).

Faça um construtor para inicializar os atributos da classe.

Escreva um método para exibir os dados do carro e outro para retornar o valor do consumo.

Faça o main para testar a classe criada.